

# Pragmatic Unit Testing – CheatSheet 1.1

## Right-BICEP

<b>R</b>	Are the results <b>right</b> ?	Use data files, golden bits, oracle...
<b>B</b>	Are all <b>boundary</b> conditions CORRECT	Conformance, <b>O</b> rdering, <b>R</b> ange, <b>R</b> eference, <b>E</b> xistence, <b>C</b> ardinality, <b>T</b> ime.
<b>I</b>	Can you check <b>inverse</b> relationships?	Apply logical inverse (e.g. sqrt vs pow).
<b>C</b>	Can you <b>cross-check</b> results using other means?	Use alternative way of achieving result.
<b>E</b>	Can you enforce <b>error conditions</b> to happen?	What errors could occur, e.g. environmental constraints.
<b>P</b>	Are <b>performance</b> characteristics within bounds?	Quick regression test of performance characteristics.

## CORRECT

<b>C</b>	Conformance	Any specific data format? What if the format is different?
<b>O</b>	Ordering	Order of data or position of an element? Reverse order?
<b>R</b>	Range	Start and end of indices, first is greater than last, index is negative, index is greater than allowed, count doesn't match number of actual number of items...
<b>R</b>	Reference	Any external dependencies? Any preconditions? Do we guarantee post-conditions?
<b>E</b>	Existence	Does some given thing exist? Null, blank, empty, 0 (zero)?
<b>C</b>	Cardinality	12 feet lawn, each section 3 feet, how many poles required? 4? Actually 5! Test for how many things there might be: Zero, one or more than one: 0-1-n-Rule.
<b>T</b>	Time	Relative time (ordering in time), absolute time (elapsed time), concurrency issues?

## A-TRIP (Good Tests)

<b>A</b>	Automatic	Running the test and checking the results should be automatic.
<b>T</b>	Thorough	Well-tested methods may have 4-5 asserts.
<b>R</b>	Repeatable	Run over and over again, in any order produce the same results.
<b>I</b>	Independent	Keep tests tight, focused, test only one thing at once.
<b>P</b>	Professional	Write real code, refactor.

